

E5994

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 11-154138

(43)Date of publication of application : 08.06.1999

(51)Int.Cl.

G06F 15/00

G06F 9/06

G06F 13/00

(21)Application number : 09-337673

(71)Applicant : FUJI XEROX CO LTD

(22)Date of filing : 21.11.1997

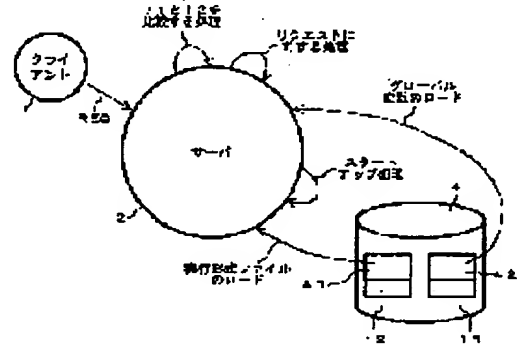
(72)Inventor : ASANO SADAJI

(54) SERVER

(57)Abstract:

PROBLEM TO BE SOLVED: To safely enable version-up of a module without having to stop a system concerning a client/server system in a distributed object environment.

SOLUTION: A global parameter and its structure information are backed up in a backup file 42 on a disk 4. The structure information of the global parameter on a module 41 is written out in the load module 41 on the disk 4 as well. At the time of start-up, two pieces of said structure information are compared. When both pieces of structure information are coincident as a result of comparison, start-up is made successful. When both pieces of structure information are not coincident by the version-up of the module, on the other hand, fail information is reported to a client or according to the structure information on the side of the module 41, the global parameter is defined again.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2000 Japanese Patent Office

This Page Blank (uspto)

(19)日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平11-154138

(43)公開日 平成11年(1999)6月8日

(51)Int.Cl.⁸

識別記号

F I

G 0 6 F 15/00
9/06
13/003 1 0
4 1 0
3 5 7G 0 6 F 15/00
9/06
13/003 1 0 C
4 1 0 Q
3 5 7 Z

審査請求 未請求 請求項の数 4 F D (全 6 頁)

(21)出願番号

特願平9-337673

(22)出願日

平成9年(1997)11月21日

(71)出願人

000005496

富士ゼロックス株式会社

東京都港区赤坂二丁目17番22号

(72)発明者

浅野 貞二

埼玉県岩槻市府内3丁目7番1号 富士ゼ

ロックス株式会社内

(74)代理人

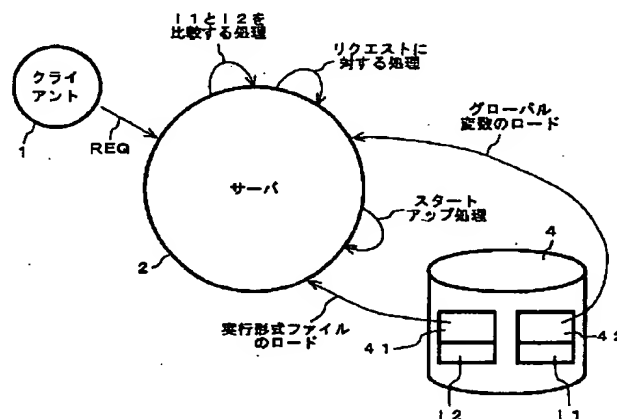
弁理士 田中 香樹 (外1名)

(54)【発明の名称】 サーバ

(57)【要約】

【課題】 分散オブジェクト環境下におけるクライアント・サーバ・システムにおいて、システムを停止させることなく安全にモジュールのバージョンアップをすることができるようになる。

【解決手段】 ディスク4上のバックアップファイル42にグローバル変数およびその構造情報をバックアップする。ディスク4上のロードモジュール41にも、該モジュール上のグローバル変数の構造情報を書き出す。スタートアップ時に、前記二つの構造情報を比較する。比較の結果、両者が一致したときはスタートアップは成功する。一方、モジュールがバージョンアップされていて両者が不一致であったならば、クライアントにフェール情報を通知するか、モジュール41側の構造情報に従ってグローバル変数を再定義する。



【特許請求の範囲】

【請求項1】 クライアント・サーバ型の分散オブジェクト環境に配置されたサーバにおいて、

サーバ上のグローバル変数の複製およびその構造情報

(第1構造情報)を格納したバックアップファイルと、サーバのアプリケーション・プログラム、ならびに該アプリケーション・プログラム上のグローバル変数の構造情報(第2構造情報)を格納した実行形式ファイルと、サーバプロセスのスタートアップ時に、前記第1構造情報および第2構造情報の一致・不一致を判別するグローバル変数の構造比較手段と、

前記第1構造情報および第2構造情報が不一致のときにスタートアップ失敗を表すフェール情報をクライアントに通知するフェール処理手段とを具備したことを特徴とするサーバ。

【請求項2】 グローバル変数の位置の変更により前記不一致が生じた場合に、該グローバル変数を前記第2構造情報に従って再配置する再配置手段を具備したことを特徴とする請求項1記載のサーバ。

【請求項3】 グローバル変数の新たな追加により前記不一致が生じた場合に、該追加されたグローバル変数を初期化する初期化手段を具備したことを特徴とする請求項1記載のサーバ。

【請求項4】 グローバル変数間の依存関係が新たに発生して前記不一致が生じた場合に、該依存関係にあるすべてのグローバル変数を初期化する初期化手段を具備したことを特徴とする請求項1記載のサーバ。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、サーバに関し、特に、分散オブジェクト環境下におけるサーバプログラムのアップデート(更新)前後においてグローバル変数が整合していない場合にグローバル変数を再構成して処理を継続することができるサーバに関する。

【0002】

【従来の技術】クライアント・サーバ型の分散オブジェクト環境下では、一般的にネットワーク上に存在する複数のホストコンピュータがクライアント・サーバ型のコンピュータシステムを構成している。このコンピュータシステムでは、各サーバは特定のホストコンピュータに依存することなく配置されていて、クライアントは目的のサーバ処理を実行するサーバが存在するホストコンピュータを意識することなく処理を依頼(リクエスト)することができる。このようなシステムでは、多数のコンピュータに分散されたオブジェクトが、代表的には、Common Object Request Broker Architecture(CORBA標準)によって統合的に管理されている。CORBA標準の環境下では、Object Request Broker(ORB)と呼ばれるシステムプログラムが各ホストコンピュータに配置されていて、クライアントとサーバ間のやりとり

を管理している。

【0003】クライアントとサーバ間のインタフェースはInterface Definition Language(IDL)と呼ばれる記述言語であらかじめ定義されていなければならない。この記述言語で定義されたインタフェースは、すべてのホストコンピュータ上のORBが解釈可能であって、この定義に従ってクライアントからサーバへのリクエストが管理される。特開平8-272725号公報には、CORBA標準による分散オブジェクト環境においてサーバの構成情報を判定し、操作するためのシステムおよび方法が開示されている。

【0004】上記コンピュータシステムにおいては、システム上に存在するすべてのサーバが、クライアントからのリクエストに即座に対応できるように常にアクティブ状態におかれていることが望ましい。しかし、このようにすべてのサーバを常にアクティブ状態に維持することは、コンピュータ上の資源を無駄に消費することとなる。

【0005】そこで、クライアントからリクエストがない場合は、ORBの管理の下、サーバのプロセスを停止したシャットダウン状態にしている。すなわち、メモリやテーブル類等の資源を解放してプロセス・コントロール・ブロックを消去している。

【0006】そして、新たなリクエストが発行されたときに、ORBは新たなプロセスをサーバにアサインし(割り当て)、サーバをアクティブ状態にする。アクティブ状態になると、サーバの記憶空間はシャットダウン前とは別の新たな記憶空間となり、先に記憶空間に存在したカウンタ値等のグローバル変数の値は保証されない。

【0007】サーバは、このような不具合に鑑み、ディスク上のファイルにグローバル変数の複製をもち、シャットダウン後のスタートアップ時に、この複製のグローバル変数を読み出すように構成している。このグローバル変数のバックアップ方式は、システムのクラッシュ、例えば、バグ等によるプロセス停止の後の復旧時にもグローバル変数のバックアップ方式として利用することができる。

【0008】分散オブジェクト環境下では、特定のサーバの保守をするときにもシステム全体を停止しなくてもよいという利点がある。例えば、モジュールつまりアプリケーション・ファイルのアップデートの際、サーバがシャットダウン状態においてはプロセスとサーバとの対応関係がなくなっていることから、ディスク上のサーバの実行形式ファイルつまりサーバのアプリケーション・プログラムを入れ替えるのみで、新たなリクエストを受けたとき、サーバはアクティブ状態になる。

【0009】

【発明が解決しようとする課題】しかし、上記バックアップされたグローバル変数を使用して新たなリクエスト

に応答する方式では、ディスク上のファイルにバックアップされたグローバル変数の構造に変更がない場合にのみ正常な起動が可能である。すなわち、モジュールのアップデートの際、新しいサーバでグローバル変数の構造がシャットダウン前から変更されている場合には、ディスク上のサーバの実行形式ファイルを入れ替えるのみでは、サーバをアクティブ状態にすることができない。

【0010】本発明は、上記問題点を解消し、モジュールのアップデートの際、シャットダウン前後においてグローバル変数の変更があった場合でも容易にスタートアップさせることができるサーバを提供することを目的とする。

【0011】

【課題を解決するための手段】上記の課題を解決し、目的を達成するための本発明は、クライアント・サーバ型の分散オブジェクト環境に配置されたサーバにおいて、サーバ上のグローバル変数の複製およびその構造情報

(第1構造情報)を格納したバックアップファイルと、サーバのアプリケーション・プログラム、ならびに該アプリケーション・プログラム上のグローバル変数の構造情報(第2構造情報)を格納した実行形式ファイルと、サーバプロセスのスタートアップ時に、前記第1構造情報および第2構造情報の一致・不一致を判別するグローバル変数の構造比較手段と、前記第1構造情報および第2構造情報が不一致のときにスタートアップ失敗を表すフェール情報をクライアントに通知するフェール処理手段とを具備した点に特徴がある。

【0012】上記特徴によれば、実行形式ファイルにアプリケーション・プログラムのグローバル変数の構造情報をもたせた。したがって、アプリケーション・プログラムのアップデートでグローバル変数の構造に変更があった場合に、そのことを、第1および第2構造情報の比較によって判別することができる。そして、グローバル変数の変更があった場合は、そのままスタートアップできないので、クライアントへスタートアップ失敗を通知してオペレータの介入を促すことができる。

【0013】

【発明の実施の形態】以下、本発明の実施形態を図面を参照して詳細に説明する。図2は、本発明の一実施例に係るクライアント・サーバ・システムの構成を示すブロック図である。同図において、LAN等のネットワーク上にワークステーションWS1、WS2、WS3が接続されている。ワークステーションWS1、WS2、WS3は、それぞれシステムプログラムとしてORBを有している。ここでは、ワークステーションWS1にクライアント1が存在し、ワークステーションWS2、WS3にサーバ2、3が存在する場合を想定している。

【0014】図3は、上記クライアント・サーバ・システムのインタフェースの構造を示す図である。同図において、ワークステーションWS1～WS3はシステムプ

ログラムとしてのORBと記述言語IDLで定義されたインタフェースを介してデータの授受をする。すなわち、ワークステーションWS1上のクライアント1で発行されたリクエストREQはIDLおよびORBを介してワークステーションWS2、WS3上のサーバ2、3に通知される。また、前記各サーバ2、3での処理結果は、IDL5およびORB6を介して前記クライアント1に通知される。

【0015】クライアント1がサーバ2に対してリクエストを発した場合のサーバ2の動作を図1を参照して説明する。サーバ2はコンピュータとこのコンピュータ上で実行されるアプリケーション・プログラムをロードするためのメモリを有する。ディスク4はアプリケーション・プログラムを格納した実行形式ファイル41とグローバル変数のバックアップファイル42とを有している。グローバル変数は、サーバ2でのサービスの実行によって変数の書き換えが発生する毎に、バックアップファイル42に格納、つまりバックアップされる。したがって、シャットダウン時には、グローバル変数が確実にバックアップされており、再度のスタートアップ時には、バックアップファイル42のグローバル変数を使用することができる。なお、バックアップファイル42にはグローバル変数のほか該変数の構造情報I1も格納されている。前記実行形式ファイル41もアプリケーション・プログラム上のグローバル変数の構造情報I2を有している。

【0016】サーバ2はリクエストREQを受けるまではシャットダウン状態である。サーバ2はリクエストREQを受けると、ORBの管理動作によりディスク4上の実行形式ファイル41をサーバ3のメモリ上にロードする。実行形式ファイル41がロードされると、サーバ2は自己のスタートアップ処理を開始する。スタートアップ処理では、例えばローカル変数や資源つまりメモリやI/Oの初期化が行われる。スタートアップ処理には、ディスク4上のバックアップファイル42に格納されているグローバル変数の複製をロードして、シャットダウン前の状態の復元を図る処理も含まれる。

【0017】サーバ2における復元処理では、必要に応じてグローバル変数の再配置が実行される。すなわち、シャットダウン状態の間にアプリケーション・プログラムがアップデートされている場合がある。この場合、シャットダウン前後でバックアップファイル42のグローバル変数の構造情報I1とアプリケーション・プログラム上のグローバル変数の構造情報I2とが異なっていることがある。そこで、グローバル変数の再配置では、グローバル変数のバックアップファイル42に含まれているグローバル変数の構造情報I1と実行形式ファイル41に含まれているグローバル変数の構造情報I2とを比較する。比較の結果、両者が一致した場合はシャットダウン前の状態が復元されたので、クライアント1からの

リクエストREQの処理を開始する。

【0018】一方、サーバ2がシャットダウン状態の間に前記実行形式ファイル41がアップデートされていて両者が一致しない場合は、グローバル変数の補正処理を行った後にリクエストREQの処理を開始する。

【0019】グローバル変数の構造情報I1、I2が一致しない場合、その不一致の態様により復元処理が異なる。以下、グローバル変数のディスク4上の位置が異なる場合と、新たなグローバル変数が追加された場合を説明する。図4(a)はバックアップされていたグローバル変数の構造情報I1の一例であり、図4(b)は実行形式ファイル41に含まれるグローバル変数の構造情報I2の一例である。構造情報I1、I2には、各グローバル変数のタイプ、サイズ、位置、および依存関係が含まれている。

【0020】図4の構造情報I1、I2から、シャットダウン前後では変数Aおよび変数Cの位置が変更されているのに加え、変数Bが追加されていることがわかる。さらに、追加された変数Bは変数に対してプログラム上で依存関係にあることも記述されている。依存関係の例としては、例えば、変数Aを算出するときに、変数Bが用いられる場合がある。したがって、このような依存関係があるときには、変数Aが初期化される時には変数Bも同時に初期化される必要がある。

【0021】なお、依存関係を含む構造情報は、実行形式ファイル41をコンパイルするコンパイラの拡張機能により生成するように構成できる。例えば、アルゴリズム上、「A. table=K+B; If A. table>4 then A. seat=TRUE」となっていれば、コンパイラの構文解析機能により図4(b)に示した依存関係が生成される。

【0022】シャットダウン前後の構造情報が異なるグローバル変数の復元処理をフローチャートを参照して説明する。図5において、スタートアップされた後、ステップS1では前記構造情報I1とI2とを比較する。ステップS2では、前記構造情報I1とI2とが一致しているか否かを判断する。一致していなかった場合は、ステップS3に進み、追加されているグローバル変数があるか否かを判断する。グローバル変数が追加されていた場合はステップS4に進み、追加されているグローバル変数を初期化する。この実施形態では変数Bが追加されていて該変数Bはバックアップファイル42からロードされることはないので、該変数Bの初期化が行われる。グローバル変数の初期化は、該変数Bのオブジェクトにおけるコンストラクタを有効にすることによって実行される。例えば、変数Bがカウンタ値である場合、該カウンタ値のインクリメントおよびデクリメントを実行するオブジェクトにおいて、該オブジェクトの生成(インストール)時にコンストラクタが有効となり、グローバル変数であるカウンタ値に「0」がセットされ、初期化さ

れる。

【0023】ステップS5では追加された変数が他のグローバル変数と依存関係にあるか否かを判断する。依存関係があればその依存関係にある変数を初期化する。この実施形態ではグローバル変数Aの要素である「table」および「seat」と依存関係があるので、これらの変数を初期化する。グローバル変数の追加がない場合、および追加があっても依存関係がない場合は、ステップS4～S6はスキップされる。

【0024】ステップS7では、グローバル変数の位置に変更があったか否かを判断する。この判断が肯定ならばステップS8に進み、グローバル変数をディスク4上に再配置する。この実施形態では、構造情報I2に従って、グローバル変数を再構成する。すなわち、変数Aとして位置(Location)1に置かれていた値は位置3に変数Aとして再配置される。同様に、変数Cとして位置3に置かれていた値は位置1に変数Cとして再配置される。こうして再配置されたグローバル変数はサーバ2のメモリへロードされる。ステップS9では、クライアント1から要求があったサービスを開始する。グローバル変数の位置変更がなかった場合は、ステップS10でグローバル変数の追加があったか否かを判断し、この判断が肯定ならば、復元処理が終了したので、ステップS9に進む。ステップS10が否定ならば追加構造情報I1、I2の不一致はないので、ステップS11に進んでフェール情報をクライアント1に通知してスタートアップ失敗で終了する。

【0025】上述の実施形態では、変数の位置の変更および変数の追加があった例を示した。本発明は、これらに限定されず、変数のタイプや変数のサイズが変更されていた場合にも同様に、該当するグローバル変数を初期化することで対応できる。

【0026】また、本実施形態では、グローバル変数の構造情報I1、I2が異なった場合、グローバル変数を再配置してロードする例を示した。しかし、これに限らず、グローバル変数の不一致を検出した場合、サービス開始のための処理を中断し、オペレータへの介入を要求するフェール情報をクライアント1に通知してもよい。これによってクライアント1がリカバリー処理に入ることができる。また、不一致となったグローバル変数が、あらかじめ定めた特定の変数である場合にのみ、前記フェール情報を通知するようにしてもよい。

【0027】以上のように、本実施形態によれば、サーバプログラムつまりサーバのアプリケーション・プログラムのアップデートにより、グローバル変数の構造に変更が生じた場合は、クライアントに通知してオペレータの介入を促すことができる。また、サーバプログラムのアップデートによるグローバル変数の構造変更に応じて自動的にグローバル変数の再定義をすることができるので、システムを停止することなく安全に実行形式ファイ

ルの入れ替えをして、サーバプログラムのアップデートをすることができる。

【0028】

【発明の効果】以上の説明から明らかなように、本発明によれば、サーバプログラムのバージョンアップ等によってグローバル変数の構造に変更が生じた場合、この構造の変更を判別して、クライアントに通知する等の適当なリカバリ処理をすることができる。

【図面の簡単な説明】

【図1】 本発明の一実施形態に係るクライアント・サーバシステムのデータの流れを示す図である。

【図2】 本発明の一実施形態に係るクライアント・サ

ーバ・システムの構成を示すブロック図である。

【図3】 分散オブジェクト環境下のクライアント・サーバ・システムのインタフェースの構造を示す図である。

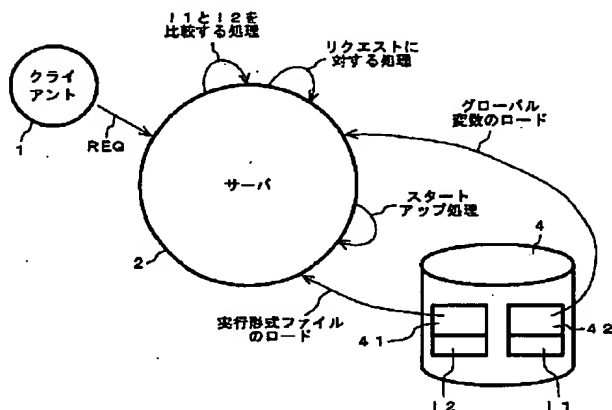
【図4】 グローバル変数の構造情報の一例を示す図である。

【図5】 グローバル変数の再定義のフローチャートである。

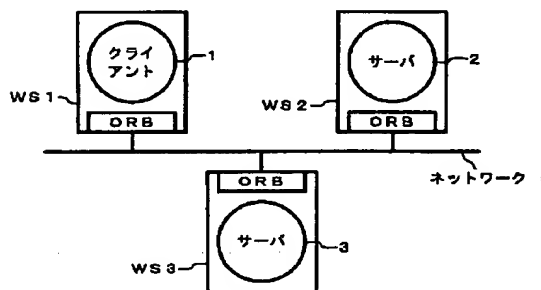
【符号の説明】

1…クライアント、 2, 3…サーバ、 4…ディスク装置、 41…実行形式ファイル、 42…バックアップファイル、 I1, I2…構成情報

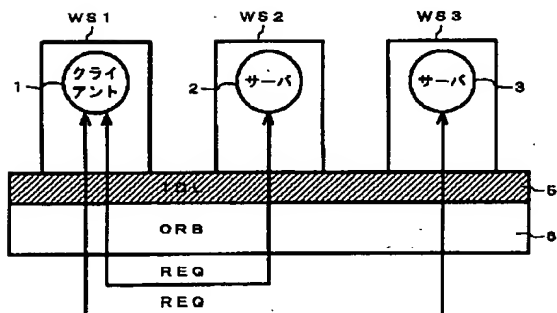
【図1】



【図2】



【図3】



【図4】

(a)

名称	タイプ	サイズ	位置	依存関係
A	A_struct	2435	Location1	
C	C_struct	200	Location3	

(b)

名称	タイプ	サイズ	位置	依存関係
A	A_struct	2435	Location3	
B	B_struct	3327	Location2	A.table, A.set
C	C_struct	200	Location1	

【図5】

